# Fundamentos das Linguagens de Programação – UFC

- \*\*01-introducao/\*\*

1957 - FORTRAN (Primeira linguagem de alto nível)

1958 - LISP (Primeira linguagem funcional)

1964 - BASIC (Linguagem para iniciantes)

1970 - Pascal (Linguagem estruturada)

1972 - C (Linguagem de sistemas influente)

1983 - C++ (Extensão OO do C)

1987 - Perl (Linguagem de script)

1991 - Python (Linguagem multiparadigma)

1995 - Java (Linguagem multiplataforma)

1995 - JavaScript (Linguagem para web)

2000 - C# (Linguagem da Microsoft)

2009 - Go (Linguagem concorrente do Google)

2010 - Rust (Linguagem de sistemas segura)

2012 - TypeScript (JavaScript tipado)

- \*\*02-ambientes/\*\*

Diagrama de Compilador vs Interpretador vs Máquina Virtual

[Fonte] → [Compilador] → [Código de Máquina]

[Fonte] → [Interpretador] → Resultado direto

[Fonte Java] → [Bytecode] → [JVM]

Exemplos:

Compilador: C, Rust

Interpretador: Python, Ruby

Máquina Virtual: Java (JVM), C# (.NET CLR)

- \*\*03-sintaxe-semantica/\*\*

Mini-Gramática: Linguagem MiniLang

Programa ::= Inicio Bloco Fim

Bloco ::= Comando ; Bloco | Comando

Comando ::= Imprima Texto | Se Condicao Entao Comando

Texto ::= "[^"]\*"

Condicao ::= Verdadeiro | Falso

Análise léxica:

Imprima "Olá mundo"

=> [Imprima] [String: "Olá mundo"]

- \*\*04-tipos-de-dados/\*\*

Comparativo entre Python, C e JavaScript:

# Python (dinâmica)

x = "10" + 5 # Erro em tempo de execução

// C (estática)

int x = "10" + 5; // Erro de compilação

// JavaScript (fraca)

let x = "10" + 5; // Resultado: "105"

- \*\*05-estruturas-de-controle/\*\*

# Sistema de recomendação de filmes

genero = input("Digite seu gênero favorito (acao/comedia/terror): ")

idade = int(input("Digite sua idade: "))

if genero == "acao":

if idade >= 18:

print("Recomendo: John Wick (2014)")

else:

print("Recomendo: Homem-Aranha (2002)")

elif genero == "comedia":

print("Recomendo: Se Beber, Não Case (2009)")

else:

print("Recomendo: O Iluminado (1980)")

# Lista de filmes assistidos

for filme in ["Matrix", "Toy Story", "It"]:

print(f"Já assistiu {filme}? (s/n)")

resposta = input()

if resposta == "s":

print("Ótima escolha!")

break

- \*\*06-subprogramas/\*\*

Python - Referência:

def alterar\_lista(lista):

lista.append(42)

C - Valor:

void soma(int a) {

a = a + 10;

}

- \*\*07-implementacao-subprogramas/\*\*

fatorial(3)

→ fatorial(2)

→ fatorial(1)

→ fatorial(0) [retorna 1]

→ retorna 1 \* 1 = 1

→ retorna 2 \* 1 = 2

→ retorna 3 \* 2 = 6

- \*\*08-orientacao-objetos/\*\*

class Personagem:

def \_\_init\_\_(self, nome): self.nome = nome

def falar(self): print(f"{self.nome} diz olá!")

class Guerreiro(Personagem):

def atacar(self): print(f"{self.nome} ataca com espada!")

- \*\*09-concorrencia/\*\*

Threads: Unidades de execução dentro de um mesmo processo (compartilham memória)

Processos: Instâncias independentes de programas (memória isolada)

Exemplo em Python:

python

import threading

def contador(inicio, fim):

for i in range(inicio, fim):

print(i)

# Duas threads no mesmo processo

t1 = threading.Thread(target=contador, args=(1, 5))

t2 = threading.Thread(target=contador, args=(11, 15))

t1.start()

t2.start()

- \*\*10-gerenciamento-memoria/\*\*

Java C

Alocação Automática (new) Manual (malloc)

Liberação Garbage Collector Manual (free)

Segurança Verificação de limites Acesso direto

Performance Overhead do GC Controle total

- \*\*11-programacao-funcional/\*\*

from functools import reduce

# Fatorial com recursão e reduce

fatorial = lambda n: 1 if n == 0 else reduce(lambda x, y: x\*y, range(1, n+1))

# Filtrar números pares e dobrá-los

numeros = [1, 2, 3, 4, 5]

resultado = list(map(lambda x: x\*2, filter(lambda x: x%2 == 0, numeros)))

# [4, 8]

- \*\*12-programacao-logica/\*\*

pai(joao, maria).

pai(joao, pedro).

mae(ana, maria).

mae(ana, pedro).

irmao(X, Y) :- pai(P, X), pai(P, Y), mae(M, X), mae(M, Y), X \= Y.

- \*\*13-scripts-web/\*\*

import os

import shutil

# Organizar arquivos por extensão

def organizar\_diretorio(diretorio):

for arquivo in os.listdir(diretorio):

if os.path.isfile(os.path.join(diretorio, arquivo)):

ext = arquivo.split('.')[-1]

pasta\_ext = os.path.join(diretorio, ext)

if not os.path.exists(pasta\_ext):

os.mkdir(pasta\_ext)

shutil.move(

os.path.join(diretorio, arquivo),

os.path.join(pasta\_ext, arquivo)

)

organizar\_diretorio('./downloads')

- \*\*14-tendencias/\*\*

Rust é uma linguagem de sistemas que combina:

Performance comparável a C/C++

Segurança de memória sem garbage collector

Concorrência sem data races

Sistema de ownership inovador

Exemplo:

rust

fn main() {

let mut s = String::from("hello");

s.push\_str(", world!");

println!("{}", s); // hello, world!

// Ownership previne referências inválidas

let s1 = String::from("Rust");

let s2 = s1; // s1 é movido para s2

// println!("{}", s1); // Erro! s1 não é mais válido

}

Principais vantagens:

Sem null pointers ou dangling pointers

Thread safety garantido em tempo de compilação

Sem overhead de runtime

Interoperabilidade com C

Ecossistema crescente (Cargo)